CameraRental Application

Phase -1 End Project

Source Code

CameraRentalApp.java (consists of main method)

```java
package phase1;
import java.util.*;
public class CameraRentalApp {
   public static void main(String[] args) {
      //CameraRentalApplication class is in camera.java file we have created a
instance to call various functions
       CameraRentalApplication app = new CameraRentalApplication();
       //User.java files consists of setting username and password method
       User u=new User();
       Scanner scanner = new Scanner(System.in);
       int choice;

       System.out.println("+---+---+----+----+----+---+----+----+");
       System.out.println("|Welcome To Camera Rental Application|");
       System.out.println("+---+---+---+---+---+---+---+----+---+");
       System.out.println("Please Login to Continue");
       System.out.println("+-------+");
       System.out.print("Username: ");
       System.out.println("\n+-------+");
       String admin=scanner.next();
       u.setName(admin);
       System.out.println("+-------+");
       System.out.print("Password: ");
       System.out.println("\n+-------+");
       String password=scanner.next();
       u.setPassword(password);

//          System.out.println(u.toString());
//                  to check whether the enter admin name and password are
returned.

       //checking whether the entered uname and pwd is crt or not
       if(admin.equalsIgnoreCase(u.getName()) &&
password.equals(u.getPassword())  ){
       do {
             //displaying the menu screen on every choice
           displayWelcomeScreen();
           choice = scanner.nextInt();
           switch (choice) {
               case 1:
                   handleAddCamera(app, scanner);
                   break;
               case 2:
                   handleRentCamera(app, scanner);
                   break;
               case 3:
                   handleWalletManagement(app, scanner);
                   break;
               case 4:
                   app.displayCameraList();
                  break;
               case 5:
                   System.out.println("Enter the camera brand:");
                   String brand=scanner.next();
                   System.out.println("Enter the camera model");
                   String model=scanner.next();
                   app.search(brand, model);
                  break;
               case 6:
                   System.out.println("Exiting the application. Goodbye!");
                   break;
               default:
                   System.out.println("Invalid choice. Please try again.");
           }
       } while (choice != 6);
   }
```

```
        else {
            System.out.print("You have entered the Wrong password or username");
        }

}

   private static void displayWelcomeScreen() {
        System.out.println("---+---+----+----+----+---+");
        System.out.println("Camera Rental Application Main Menu");
        System.out.println("---+---+---+---+---+---+---+");
        System.out.println("1. Add a camera");
        System.out.println("2. Rent a camera");
        System.out.println("3. Wallet Management");
        System.out.println("4. Display Camera List");
        System.out.println("5. Want to search any camera?");
        System.out.println("6. Exit");
        System.out.print("Enter your choice: ");
   }
   //method to add camera , inside which called another method from camera.java
   private static void handleAddCamera(CameraRentalApplication app, Scanner
scanner) {

       int choice;

       do {
            System.out.println("---+---+---+---+---+---+---+");
            System.out.println("1. Add a camera");
            System.out.println("2. Remove");
            System.out.println("3. My cameras ");
            System.out.println("4. Back to main menu");
            System.out.print("Enter your choice: ");
             choice=scanner.nextInt();
            System.out.println("\n---+---+---+---+---+---+---+");
             switch(choice) {
             case 1:
             System.out.println("Add a Camera");
                System.out.println("------------");
                scanner.nextLine(); // Consume newline character
                System.out.print("Enter the brand: ");
                String brand = scanner.nextLine();
                System.out.print("Enter the model: ");
                String model = scanner.nextLine();
                System.out.print("Enter the per-day rental amount: ");
                double rentalAmount = scanner.nextDouble();
                app.addCamera(brand, model, rentalAmount);
                System.out.println("Camera added successfully.");
                break;
             case 2:
                    System.out.println("Enter the index Number to Remove a 
camera:");
                    int index=scanner.nextInt();
                    try {
                                app.deleteCamera(index-1);
                          System.out.println("Camera at "+index+"Removed.");
                          } catch (InvalidIndex e) {
                                System.out.println(e.getMessage());}
                    break;
             case 3:
                    app.displayCameraList();
                    break;
             default:
                System.out.println("Enter a Valid choice:");
             }
        }while(choice!=4);
    }
    //to rent a camera and to check if the balance is available or not
    private static void handleRentCamera(CameraRentalApplication app, Scanner 
scanner) {
        System.out.println("Rent a Camera");
        System.out.println("-------------");
        app.displayCameraList();
        if (app.cameraList.isEmpty()) {
            System.out.println("No cameras available for rent.");
            return;
        }
        System.out.print("Enter the index of the camera to rent: ");
        int cameraIndex = scanner.nextInt();
        System.out.print("Enter the rental duration (in days): ");
        int rentalDuration = scanner.nextInt();
        try {
            app.rentCamera(cameraIndex-1, rentalDuration);
        } catch (InsufficientBalanceException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    //wallet prices to show the amount left and also to add the amount
    private static void handleWalletManagement(CameraRentalApplication app,
Scanner scanner) {
        System.out.println("Wallet Management");
        System.out.println("-----------------");
        System.out.println("1. View Wallet Balance");
        System.out.println("2. Deposit Funds");
        System.out.print("Enter your choice: ");
        int choice = scanner.nextInt();
        switch (choice) {
            case 1:
                app.displayWalletBalance();
                break;
            case 2:
                System.out.print("Enter the amount to deposit: ");
                double amount = scanner.nextDouble();
                app.depositToWallet(amount);
                break;
            default:
                System.out.println("Invalid choice. Returning to the main
menu.");
        }
    }
}
```

<u>Camera.java(consists of various operations to be performed)</u>

```
package phase1;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;


class Camera {
    private String brand;
    private String model;
    private double perDayRentalAmount;
    private boolean available;

    public Camera( String brand, String model, double perDayRentalAmount) {
        this.brand = brand;
        this.model = model;
        this.perDayRentalAmount = perDayRentalAmount;
        this.available = true;
    }

    // Getters and setters

    public String getBrand() {
        return brand;
    }

    public String getModel() {
        return model;
    }

    public double getPerDayRentalAmount() {
        return perDayRentalAmount;
    }

    public boolean isAvailable() {
        return available;
    }

    public void setAvailable(boolean available) {
        this.available = available;
    }
}

//custom exception
class InsufficientBalanceException extends Exception {
      private static final long serialVersionUID = 1L;
      public InsufficientBalanceException(String message) {
        super(message);
    }
}
class InvalidIndex  extends Exception {
      private static final long serialVersionUID = 1L;
      public InvalidIndex(String message) {
            super(message);
      }
}


//this is class  we used many functions such as  append a new camera to the
list and rent a camera and add or deduct money from the wallet.
class CameraRentalApplication {
     List<Camera> cameraList;
    private double walletBalance;

    public CameraRentalApplication() {
        cameraList = new ArrayList<>();
        walletBalance = 0.0;
    }

    public void addCamera(String brand, String model, double
perDayRentalAmount) {
        Camera camera = new Camera(brand, model, perDayRentalAmount);
        cameraList.add(camera);
    }


    public void deleteCamera(int index)  throws InvalidIndex{
      if(index<0) throw new InvalidIndex("Invalid Index");
      if(cameraList.isEmpty()) System.out.println("There are no cameras to
remove, You can add a camera");
      cameraList.remove(index);
    }
    //displaying the camera's present
    public void displayCameraList() {
        if (cameraList.isEmpty()) {
            System.out.println("No Data Present at This Moment.");
        } else {
            System.out.println("Available Cameras:");
            for (Camera camera : cameraList) {
                if (camera.isAvailable()) {
                    System.out.println("Brand: " + camera.getBrand());
                    System.out.println("Model: " + camera.getModel());
                    System.out.println("Per-day Rental Amount: $" +
camera.getPerDayRentalAmount());
                    System.out.println("\n--------+-+---------");
                }
            }
        }
    }

    public void rentCamera(int cameraIndex, int rentalDuration) throws
InsufficientBalanceException {
        Camera camera = cameraList.get(cameraIndex);

        if (!camera.isAvailable()) {
            System.out.println("Camera is not available for rent.");
            return;
        }

        double rentalCost = camera.getPerDayRentalAmount() * rentalDuration;

        if (walletBalance < rentalCost) {
            throw new InsufficientBalanceException("Insufficient balance in the
wallet.");
        }
        walletBalance -= rentalCost;
        camera.setAvailable(false);

        System.out.println("Camera rented successfully.");
    }

    public void displayWalletBalance() {
        System.out.println("Wallet Balance: $" + walletBalance);
    }

    public void depositToWallet(double amount) {
        if (amount <= 0) {
            System.out.println("Invalid deposit amount.");
            return;
        }

        walletBalance += amount;
        System.out.println("Deposit successful.");
    }
    //sorting
    public void sortCameraList(Comparator<Camera> comparator) {
        Collections.sort(cameraList, comparator);
    }
    //searching
      public void search(String model,String brand) {
      if (cameraList.isEmpty()) {
            System.out.println("No Data Present at This Moment.");
        } else {
      for(Camera i :cameraList) {
            if(i.getBrand().equalsIgnoreCase(brand) &&
i.getModel().equalsIgnoreCase(model)) {
                System.out.println("Camera your were looking :");
                  System.out.println("Brand: " + i.getBrand());
                System.out.println("Model: " + i.getModel());
                System.out.println("\n--------+-+---------");
            }
      }
    }
    }
}
```

User.java

```
package phase1;
public class User {
	String name;
	String password;

	//getters and setters

	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}
	@Override
	public String toString() {
		return "User [name=" + name + ", password=" + password + "]";
	}
}
```